```
                        title   'Track Buffered BIOS - SBC204'

                    ; 5 Jan 82  -jrp


                    ;       Copyright (C) 1980,1981
                    ;       Digital Research, Inc.
                    ;       Box 579, Pacific Grove
                    ;       C  ifornia, 93950
                    ;
                    ;       (Permission is hereby granted to use
                    ;       or abstract the following program in
                    ;       the implementation of CP/M, MP/M or
                    ;       CP/NET for the 8086 or 8088 Micro-
                    ;       pr :essor)


FFFF                true            equ -1
0000                false           equ not true

                    ;       Track buffering equates...

0080                host_sectsiz    equ     128
001A                host_spt        equ     26
0001                host_fsn        equ     1

00E0                bdos_int        equ 224
2500                bios_code       equ 2500h
0000                ccp_offset      equ 0000h
0B06                bdos_ofst       equ 0B06h ;BDOS entry point

00DA                csts            equ 0DAh  ;i8251 status port
00D8                cdata           equ 0D8h  ;   '  data port

0041                lsts            equ 41h ;2651 No. 0 on BLC8538 status port
0040                ldata           equ 40h ;  '   '   '  '    '    data port
0060                blc_reset       equ 60h ;reset selected USARTS on BLC8538


                    ;*********************************************
                    ;*                                           *
                    ;*    Intel iSBC 204 Disk Controller Ports   *
                    ;*                                           *
                    ;*********************************************

00A0                base204         equ 0a0h        ;SBC204 assigned address

00A0                fdc_com         equ base204+0   ;8271 FDC out command
00A0                fdc_stat        equ base204+0   ;8271 in status
00A1                fdc_parm        equ base204+1   ;8271 out parameter
00A1                fdc_rslt        equ base204+1   ;8271 in result
00A2                fdc_rst         equ base204+2   ;8271 out reset
```

CP/M 86 V.1.1-vol I
COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. # C81-162

```
                                                                    
00A4                dmac_adr        equ base204+4   ;8257 DMA base address out
00A5                dmac_cont       equ base204+5   ;8257 out control
00A6                dmac_scan       equ base204+6   ;8257 out scan control
00A7                dmac_sadr       equ base204+7   ;8257 out scan address
00A8                dmac_mode       equ base204+8   ;8257 out mode
00A8                dmac_stat       equ base204+8   ;8257 in status
00A9                fdc_sel         equ base204+9   ;FDC select port (not used)
00AA                fdc_segment     equ base204+10  ;segment address register
00AF                reset_204       equ base204+15  ;reset entire interface
                                                    
000A                max_retries     equ 10          ;max retries on disk i/o
                                                    ;before perm error
000D                cr              equ 0dh         ;carriage return
000A                lf              equ 0ah         ;line feed
                                                    
                            cseg
                            org     ccpoffset
                        ccp:
                            org     bios_code
                                                    
                        ;***********************************************
                        ;*                                             *
                        ;* BIOS Jump Vector for Individual Routines    *
                        ;*                                             *
                        ;***********************************************
                                                    
2500 E93C00         253F    jmp INIT        ;Enter from BOOT ROM or LOADER
2503 E98700         258D    jmp WBOOT       ;Arrive here from BDOS call 0
2506 E99300         259C    jmp CONST       ;return console keyboard status
2509 E99900         25A5    jmp CONIN       ;return console keyboard char
250C E9A000         25AF    jmp CONOUT      ;write char to console device
250F E9A800         25BA    jmp LISTOUT     ;write character to list device
2512 E9BA00         25CF    jmp PUNCH       ;write character to punch device
2515 E9B700         25CF    jmp READER      ;return char from reader device
2518 E9FA00         2615    jmp HOME        ;move to trk 00 on cur sel drive
251B E9DE00         25FC    jmp SELDSK      ;select disk for next rd/write
251E E90301         2624    jmp SETTRK      ;set track for next rd/write
2521 E90501         2629    jmp SETSEC      ;set sector for next rd/write
2524 E90701         262E    jmp SETDMA      ;set offset for user buff (DMA)
2527 E91F01         2649    jmp READ        ;read a 128 byte sector
252A E92F01         265C    jmp WRITE       ;write a 128 byte sector
252D E99400         25C4    jmp LISTST      ;return list status
2530 E90501         2638    jmp SECTRAN     ;xlate logical->physical sector
2533 E9FD00         2633    jmp SETDMAB     ;set seg base for buff (DMA)
2536 E90C01         2645    jmp GETSEGT     ;return offset of Mem Desc Table
2539 E99600         25D2    jmp GETIOBF     ;return I/O map byte (IOBYTE)
253C E99600         25D5    jmp SETIOBF     ;set I/O map byte (IOBYTE)
                                                    
                                                    
                                                    
                                                    ;print signon message and initialize hardware
                        INIT:
```

COPYRIGHT © 1981, 198
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 939
SER. #_____

```
                                mov ax,cs       ;we entered with a JMPF so use
                                mov ds,ax       ; CS: as the initial value of DS:,
                                mov es,ax       ;       and ES:
                                mov ss,ax       ; use local stack during initialization
                                mov sp,offset stkbase
                                cld             ;set forward direction
                                push ds         ;save the DS register
                                mov ax,0
                                mov ds,ax
                                mov es,ax       ;set ES and DS to zero
                                ;setup interrupt 0 to address trap routine
                                mov int0_offset,offset int_trap
                                mov int0_segment,CS
                                mov di,4
                                mov si,0        ;then propagate
                                mov cx,510      ;trap vector to
                                rep movs ax,ax  ;all 256 interrupts
                                ;BDOS offset to proper interrupt
                                mov bdos_offset,bdos_ofst
                                pop ds          ;restore the DS register

                        ; Initialize the BLC 8538 printer port
                                mov al,0FFh
                                out blc_reset,al ;reset all usarts on 8538
                                mov al,4Eh
                                out ldata+2,al  ;set usart 0 in async 8 bit mode
                                mov al,3Eh
                                out ldata+2,al  ;set usart 0 to 9600 baud
                                mov al,37h
                                out ldata+3,al  ;enable Tx/Rx, and set up RTS,DTR
                                mov bx,offset signon
                                call pmsg       ;print signon message

                                call clear_flags ; initialize track buffering

                                mov cl,0        ;default to dr A: on coldstart
                                jmp ccp         ;jump to cold start entry of CCP

0006 WBOOT:                     jmp ccp+6       ;direct entry to CCP at command level


                        int_trap:
                                cli             ;block interrupts
                                mov ax,cs
                                mov ds,ax       ;get our data segment
                                mov bx,offset int_trp
                                call pmsg
                                hlt             ;hardstop


                        ;***********************************************
                        ;*                                             *
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                        ;*   CP/M Character I/O Interface Routines   *
                        ;*   Console is Usart (i8251a) on iSBC 86/12 *
                        ;*   at ports D8/DA                          *
                        ;*                                           *
                        ;*********************************************

                        CONST:          ;console status
259C E4DA                       in al,csts
259E 2402                       and al,2
25A0 7402               25A4    jz const_ret
25A2 0CFF                       or al,255       ;return non-zero if RDA
                        const_ret:
25A4 C3                         ret             ;Receiver Data Available

                        CONIN:                  ;console input
25A5 E8F4FF             259C    call const
25A8 74FB               25A5    jz CONIN        ;wait for RDA
25AA E4D8                       in al,cdata
25AC 247F                       and al,7fh      ;read data and remove parity bit
25AE C3                         ret

                        CONOUT:         ;console output
25AF E4DA                       in al,csts
25B1 2401                       and al,1        ;get console status
25B3 74FA               25AF    jz CONOUT       ;wait for TBE
25B5 8AC1                       mov al,cl
25B7 E6D8                       out cdata,al    ;Transmitter Buffer Empty
25B9 C3                         ret             ;then return data

                        LISTOUT:                ;list device output

25BA E80700             25C4    call LISTST
25BD 74FB               25BA    jz LISTOUT      ;wait for printer not busy
25BF 8AC1                       mov al,cl
25C1 E640                       out ldata,al    ;send char to TI 810
25C3 C3                         ret

                        LISTST:                 ;poll list status

25C4 E441                       in al,lsts
25C6 2481                       and al,81h      ;look at both TxRDY and DTR
25C8 3C81                       cmp al,81h
25CA 750A               25D6    jnz zero_ret    ;either false, printer is busy
25CC 0CFF                       or al,255       ;both true, LPT is ready
25CE C3                         ret

                        PUNCH:  ;not implemented in this configuration
                        READER:
25CF B01A                       mov al,1ah
25D1 C3                         ret             ;return EOF for now

                        GETIOBF:
```

COPYRIGHT © 1981, 1
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93
SER #_____

```
                                mov al,0        ;TTY: for consistency
25D4 C3                         ret             ;IOBYTE not implemented

                        SETIOBF:
25D5 C3                         ret             ;iobyte not implemented

                        zero_ret:
25D6 2400                       and al,0
25D8 C3                         ret             ;return zero in AL and flags

                        ; Routine to get and echo a console character
                        ;       and shift it to upper case

                        uconecho:
25D9 E8C9FF     25A5            call CONIN      ;get a console character
25DC 50                         push ax
25DD 8AC8                       mov cl,al       ;save and
25DF E8CDFF     25AF            call CONOUT
25E2 58                         pop ax          ;echo to console
25E3 3C61                       cmp al,'a'
25E5 7206       25ED            jb uret         ;less than 'a' is ok
25E7 3C7A                       cmp al,'z'
25E9 7702       25ED            ja uret         ;greater than 'z' is ok
25EB 2C20                       sub al,'a'-'A'  ;else shift to caps
                        uret:
25ED C3                         ret

                        ;       utility subroutine to print messages

                        pmsg:
25EE 8A07                       mov al,[BX]     ;get next char from message
25F0 84C0                       test al,al
25F2 7420       2614            jz return       ;if zero return
25F4 8AC8                       mov CL,AL
25F6 E8B6FF     25AF            call CONOUT     ;print it
25F9 43                         inc BX
25FA EBF2       25EE            jmps pmsg       ;next character and loop


                        ;***********************************************
                        ;*                                             *
                        ;*         Disk Input/Output Routines          *
                        ;*                                             *
                        ;***********************************************

                        SELDSK:         ;select disk given by register CL
25FC 880E012A                   mov cpm_disk,cl                 ; save the selected drive
2600 BB0000                     mov bx,0000h
2603 80F902                     cmp cl,2        ;this BIOS only supports 2 disks
2606 730C       2614            jnb return      ;return w/ 0000 in BX if bad drive
                                                ;now, we need disk parameter address
```

COPYRIGHT © 1981, 1
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93
SER. #_____

```
                                mov ch,0
2608 B500                       mov bx,cx       ;BX = word(CL)
260A 8BD9                       mov cl,4
260C B104                       shl bx,cl       ;multiply drive code * 16
260E D3E3                       add bx,offset dp_base   ;create offset from Disk Parameter Base
2610 81C31A2A           return:
2614 C3                         ret

                        home:
2615 F606102A0175 2621          test wr_flag,1 ! jnz home1 ; if the buffer is clean,
     05           2621
261C C606062AFF                 mov     cur_disk,-1     ; insure we read the directory by invalidating
                                                        ;  the track buffer
                        home1:
2621 B70000                     mov     cx,0            ; home is a settrk zero

                        settrk:
2624 890E022A                   mov     cpm_track,cx    ; save track number for next operation
2628 C3                         ret

                        setsec:
2629 890E042A                   mov     cpm_sec,cx      ; save sector number for next operation
262D C3                         ret


                        setdma:
262E 890E0B2A                   mov     dma_offset,cx           ; save DMA offset address
2632 C3                         ret
                        setdmab:
2633 890E0D2A                   mov     dma_segment,cx          ; save DMA segment address
2637 C3                         ret

                        sectran:
2638 8BD9                       mov     bx,cx           ; Put logical sector into dest. reg.
263A 85D2                       test    dx,dx           ; see if table address is zero
263C 7406               2644    jz      sectran_exit    ; yeah, logical = physical
263E 03DA                       add     bx,dx           ; else, we need to fetch the
2640 8A1F                       mov     bl,[BX]         ;  actual sector number from the table
2642 B700                       mov     bh,0            ; zero high byte for good luck
                        sectran_exit:
2644 C3                         ret


                        GETSEGT:  ;return address of physical memory table
2645 BB112A                     mov bx,offset seg_table
2648 C3                         ret


                        read:
2649 E84800             2694    call    track_setup
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                                
264C 06                         push    es                      ; save the extra segment register
264D BE5B2B                     mov     si,offset track_buffer  ; source segment is systems DS:
2650 03F0                       add     si,ax                   ; gives the offset into the buffer
2652 C43E0B2A                   les     di,dma_longword         ; point ES:DI at the users sector
2656 F3A5                       rep movsw                       ; doit
2658 07                         pop     es                      ; restore the extra segment
2659 2BC0                       sub     ax,ax                   ; make a zero return code
265B C3                         ret
                                
                        write:
265C 51                         push    cx                      ; save the write mode from the BDOS
265D E83400             2694    call    track_setup
2660 50                         push    ax                      ; save buffer offset
2661 1E                         push    ds                      ; save the data segment
2662 06                         push    es                      ; save the extra segment
2663 8CD88EC3                   mov bx,ds ! mov es,bx           ; destination is our data segment
2667 BF5B2B                     mov     di,offset track_buffer  ; destination is in track buffer
266A 03F8                       add     di,ax                   ; plus appropriate offset
266C C5360B2A                   lds     si,dma_longword         ; source is users DMA address
2670 F3A5                       rep movsw                       ; move that sector
2672 07                         pop     es                      ; restore the extra segment
2673 1F                         pop     ds                      ; and the data segment registers
2674 58                         pop     ax                      ; recover buffer offset
2675 B98000                     mov     cx,host_sectsiz         ; setup to divide by host sector size
2678 2BD2                       sub     dx,dx                   ; extend ax to 32 bits
267A F7F1                       div     cx                      ; find out which host sector we changed
267C 8BD8                       mov     bx,ax                   ; put into index [BX]
267E C687412B01                 mov     sec_flags[BX],1         ; set the update flag for that sector
2683 C606102A01                 mov     wr_flag,1               ; also set the dirty buffer flag
2688 59                         pop     cx                      ; recover BDOS write code
2689 80F901                     cmp     cl,1                    ; is this a directory update ?
268C 7503               2691    jne     write_return            ; no, we may leave dirty records
                                                                ;       in the buffer
268E E84200             26D3    call    flush_buffer            ; we have a directory write, need to
                                                                ; flush the buffer to insure the
                                                                ; disks integrity
                        write_return:
2691 2BC0                       sub     ax,ax                   ; never return BAD SECTOR code
2693 C3                         ret
                                
                                
                        track_setup:                    ; common code for setting up reads and writes
                                
2694 A0012A                     mov     al,cpm_disk             ; see if selected disk is
2697 3A06062A                   cmp     al,cur_disk             ;       the same as last time
269B 7509               26A6    jne     wrong_track             ; no, we have wrong track
                                
269D A1022A                     mov     ax,cpm_track            ; see if desired track is same as
26A0 3B06072A                   cmp     ax,cur_track            ;       the track in the buffer
26A4 741E               26C4    je      correct_track           ; same drive and track, we don't need to read
                                
                        ;       Desired operation is on a different track then is in our buffer,
```

```
372
373                             ;       So it will be nessecary to read in the desired track.  First, we
374                             ;       must check to see if any sectors of the current buffer are dirty.
375
376                             wrong_track:
377  26A6 E82A00        26D3            call    flush_buffer            ; write any old records, if nessecary
378
379  26A9 A1022A                        mov     ax,cpm_track            ; get desired track number
380  26AC A3072A                        mov     cur_track,ax            ; make in new track
381  26AF A0012A                        mov     al,cpm_disk             ; get desired disk number
382  26B2 A2062A                        mov     cur_disk,al             ; make it current drive
383  26B5 C706F2295B2B                  mov     cur_dma_adr,offset track_buffer
384                                                                     ; point dma offset at track buffer
385  26BB C706092A0100                  mov     cur_sec,host_fsn        ; starting from first sector
386  26C1 E86800        272C            call    track_read              ; load the track
387
388                             correct_track:
389  26C4 A1042A                        mov     ax,cpm_sec              ; get the cp/m sector number
390                               if (host_fsn ne 0)
391  26C7 2D0100                        sub     ax,host_fsn             ; correct if we start with sector one
392                                 endif
393  26CA B107                          mov     cl,7                    ; log2(128)
394  26CC D3E0                          shl     ax,cl                   ; sector times 128 gives offset
395  26CE B94000FC                      mov cx,64 ! cld                 ; move 64 words forward
396  26D2 C3                            ret
397
398
399                             flush_buffer:
400  26D3 F606102A01                    test    wr_flag,1               ; see if we have anything to write
401  26D8 742E          2708            jz      no_flush                ; no, skip scanning for dirty sectors
402  26DA BB0100                        mov     bx,host_fsn             ; start at first host sector
403  26DD B91A00                        mov     cx,host_spt             ; for host_spt sectors...
404                             next_sect:
405  26E0 F687402B01                    test    sec_flags-host_fsn[BX],1 ; see if this sector has been changed
406  26E5 741E          2705            jz      not_updated              ; no, leave it alone
407  26E7 C687402B00                    mov     sec_flags-host_fsn[BX],0 ; zero the flag for next time
408  26EC 53                            push    bx                       ; save the registers
409  26ED 51                            push    cx
410  26EE 891E092A                      mov     cur_sec,bx              ; save host sector number
411  26F2 B88000                        mov     ax,host_sectsiz
412                                 if (host_fsn ne 0)
413  26F5 83EB01                        sub     bx,host_fsn
414                                 endif
415  26F8 F7E3                          mul     bx                      ; make track buffer offset
416  26FA 055B2B                        add     ax,offset track_buffer  ; make direct pointer
417  26FD A3F229                        mov     cur_dma_adr,ax          ; save for write routine
418  2700 E83500        2738            call    sector_write
419  2703 59                            pop     cx
420  2704 5B                            pop     bx
421                             not_updated:
422  2705 43                            inc     bx
423  2706 E2D8          26E0            loop    next_sect
424                             no_flush:
```

P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                
2708 C606102A00                 mov     wr_flag,0               ; clear the dirty buffer flag
270D C3                         ret
                                
                        clear_flags:    ; Clear all variables associated with the track buffer,
                                        ; so next operation will have to read a track.
                                        ;  This is involves clearing all write flags and setting
                                        ;  the old drive code to the invalid -1.
                                
270E 06                         push es                         ; save extra segment
270F C606062A0FF                mov cur_disk,-1                 ; insure initial pre-read
2714 2BC0                       sub ax,ax                       ; make a zero
2716 A2102A                     mov wr_flag,al                  ; clear the dirty buffer flag
2719 BF412B                     mov di,offset sec_flags         ; point to the update flag list
271C 8CDB8EC3                   mov bx,ds ! mov es,bx           ; ES <- DS
2720 B91A00FC                   mov cx,host_spt ! cld           ; set length and direction
2724 F3AA                       rep stosb                       ; zero the sector update flags
2726 8C1EF429                   mov cur_dma_seg,ds              ; get our segment address
272A 07                         pop es                          ; recover extra segment
272B C3                         ret
                                
                                
                                
                        track_READ:
272C C606F72904                 mov io_com,4            ; read track takes 4 byte command
2731 C606F82913                 mov io_com+1,13h        ; special read track command
2736 EB0A       2742            jmps r_w_common
                                
                        sector_WRITE:
2738 C606F72903                 mov io_com,3            ; write sector takes 3 byte command
273D C606F8290A                 mov io_com+1,0ah        ; basic write sector command
                                
                        r_w_common:
2742 803E062A0175 2758          cmp cur_disk,1 ! jne not_first_b        ; see if drive B
     0F           2758
2749 F606EE29FF74 2758          test b_first_flag,-1 ! jz not_first_b   ; and first reference to B
     03           2758
2750 E8B200       2805          call restore                            ; then restore drive B
2753 C606EE2900                 mov b_first_flag,0                      ; and clear flag
                        not_first_b:
2758 BBF729                     mov bx,offset io_com ;point to command string
275B A1092A                     mov ax,cur_sec  ; put sector in
275E A2FA29                     mov sect,al     ; iopb as 8 bits
2761 A1072A                     mov ax,cur_track ; same with
2764 A2F929                     mov trk,al      ;    track . .
                                
                        ;       fall into execute and return
                                
                        execute:  ;execute command string,
                                ;[BX] points to length,
                                ;       followed by Command byte,
                                ;       followed by length-1 parameter bytes
                                
```

COPYRIGHT (C) 1981, 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER #

```
                                
2767 891EF029                   mov last_com,BX ;save command address for retries
                        outer_retry:
                                ;allow some retrying
276B C606EF290A                 mov rtry_cnt,max_retries
                        retry:
2770 8B1EF029                   mov BX,last_com
2774 E89F00     2816            call send_com   ;transmit command to i8271
                        ;       check status poll
                                
2777 8B1EF029                   mov BX,last_com
277B 8A4701                     mov al,1[bx]    ;get command op code
277E B90008                     mov cx,0800h    ;mask if it will be 'int req'
2781 3C2C                       cmp al,2ch
2783 720B       2790            jb exec_poll    ;ok if it is an interrupt type
2785 B98080                     mov cx,8080h    ;else we use 'not command busy'
2788 240F                       and al,0fh
278A 3C0C                       cmp al,0ch      ;unless there isn't
278C B000                       mov al,0
278E 773B       27CB            ja exec_exit    ;       any result
                                                ;poll for bits in CH,
                        exec_poll:              ;  toggled with bits in CL
                                
2790 E4A0                       in al,fdc_stat  ;read status
2792 22C5                       and al,ch
2794 32C1                       xor al,cl       ;  isolate what we want to poll
2796 74F8       2790            jz exec_poll    ;and loop until it is done
                                
                                                ;Operation complete,
2798 E4A1                       in al,fdc_rslt  ; see if result code indicates error
279A 241E                       and al,1eh
279C 742D       27CB            jz exec_exit    ;no error, then exit
                                                ;some type of error occurred . . .
279E 3C10                       cmp al,10h
27A0 742A       27CC            je dr_nrdy      ;was it a not ready drive ?
                                                ;no,
                        dr_rdy: ; then we just retry read or write
27A2 50                         push ax         ; save error code
27A3 E85F00     2805            call restore    ; after physically homing this disk
27A6 58                         pop ax          ; recover error code
27A7 FE0EEF29                   dec rtry_cnt
27AB 75C3       2770            jnz retry       ;  up to 10 times
                                
                        ;       retries do not recover from the
                        ;       hard error
                                
27AD B400                       mov ah,0
27AF 8BD8                       mov bx,ax       ;make error code 16 bits
27B1 8B9FB428                   mov bx,errtbl[BX]
27B5 E836FE     25EE            call pmsg       ;print appropriate message
27B8 E4D8                       in al,cdata     ;flush usart receiver buffer
27BA E81CFE     25D9            call uconecho   ;read upper case console character
27BD 3C43                       cmp al,'C'
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER #: _____

```
                                    ;cancel
27BF 7425       27E6        je wboot_1
27C1 3C52                   cmp al,'R'
27C3 74A6       276B        je outer_retry  ;retry 10 more times
27C5 3C49                   cmp al,'I'
27C7 741A       27E3        je z_ret        ;ignore error
27C9 0CFF                   or al,255       ;set code for permanent error
                exec_exit:
27CB C3                     ret

                dr_nrdy:        ;here to wait for drive ready
27CC E81A00     27E9        call test_ready
27CF 759F       2770        jnz retry       ;if it's ready now we are done
27D1 E81500     27E9        call test_ready
27D4 759A       2770        jnz retry       ;if not ready twice in row,
27D6 BB5929                 mov bx,offset nrdymsg
27D9 E812FE     25EE        call pmsg       ;"Drive Not Ready"
                nrdy01:
27DC E80A00     27E9        call test_ready
27DF 74FB       27DC        jz nrdy01       ;now loop until drive ready
27E1 EB8D       2770        jmps retry      ;then go retry without decrement
                zret:
27E3 2400                   and al,0
27E5 C3                     ret             ;return with no error code

                wboot_1:                    ;can't make it w/ a short leap
27E6 E9A4FD     258D        jmp WBOOT

                ;***********************************************
                ;*                                             *
                ;*  The i8271 requires a read status command   *
                ;*  to reset a drive-not-ready after the       *
                ;*  drive becomes ready                        *
                ;*                                             *
                ;***********************************************

                test_ready:
27E9 B640                   mov dh, 40h     ;proper mask if dr 1
27EB F606F62980             test sel_mask,80h
27F0 7502       27F4        jnz nrdy2
27F2 B604                   mov dh, 04h     ;mask for dr 0 status bit
                nrdy2:
27F4 BBFF29                 mov bx,offset rds_com
27F7 E81C00     2816        call send_com
                dr_poll:
27FA E4A0                   in al,fdc_stat  ;get status word
27FC A880                   test al,80h
27FE 75FA       27FA        jnz dr_poll     ;wait for not command busy
2800 E4A1                   in al,fdc_rslt  ;get "special result"
2802 34C6                   test al,dh      ;look at bit for this drive
2804 C3                     ret             ;return status of ready
```

COPYRIGHT 1981,1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 584
 585                            restore:        ;move selected disk to home position (Track 0)
 586   2805 BBFC29                      mov bx,offset hom_com
 587   2808 E85CFF       2767           call execute
 588   280B 7408         2815           jz restore_exit ;home drive and return if OK
 589   280D BB8D28                      mov bx,offset bad_hom   ;else print
 590   2810 E8D8FD       25EE           call pmsg       ;"Home Error"
 591   2813 EBF0         2805           jmps restore    ;and retry
 592                            restore_exit:
 593   2815 C3                          ret
 594
 595                            ;***********************************************
 596                            ;*                                             *
 597                            ;*  Send_com sends a command and parameters    *
 598                            ;*  to the i8271:  BX addresses parameters.    *
 599                            ;*  The DMA controller is also initialized     *
 600                            ;*  if this is a read or write                 *
 601                            ;*                                             *
 602                            ;***********************************************
 603
 604                            send_com:
 605   2816 E4A0                        in al,fdc_stat
 606   2818 A880                        test al,80h     ;insure command not busy
 607   281A 75FA         2816           jnz send_com    ;loop until ready
 608
 609                                    ;see if we have to initialize for a DMA operation
 610
 611   281C 8A4701                      mov al,1[bx]    ;get command byte
 612   281F 3C13                        cmp al,13h
 613   2821 7504         2827           jne write_maybe ;if not a read it could be write
 614   2823 B140                        mov cl,40h
 615   2825 EB06         282D           jmps init_dma   ;is a read command, go set DMA
 616                            write_maybe:
 617   2827 3C0A                        cmp al,0ah
 618   2829 7520         284B           jne dma_exit    ;leave DMA alone if not read or write
 619   282B B180                        mov cl,80h      ;we have write, not read
 620                            init_dma:
 621                            ;we have a read or write operation, setup DMA controller
 622                            ;       (CL contains proper direction bit)
 623   282D B004                        mov al,04h
 624   282F E6A8                        out dmac_mode,al    ;enable dmac
 625   2831 B000                        mov al,00
 626   2833 E6A5                        out dmac_cont,al    ;send first byte to control port
 627   2835 8AC1                        mov al,cl
 628   2837 E6A5                        out dmac_cont,al   ;load direction register
 629   2839 A1F229                      mov ax,cur_dma_adr
 630   283C E6A4                        out dmac_adr,al    ;send low byte of DMA
 631   283E 8AC4                        mov al,ah
 632   2840 E6A4                        out dmac_adr,al    ;send high byte
 633   2842 A1F429                      mov ax,cur_dma_seg
 634   2845 E6AA                        out fdc_segment,al ;send low byte of segment address
 635   2847 8AC4                        mov al,ah
 636   2849 E6AA                        out fdc_segment,al ;then high segment address
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                dma_exit:
284B 8A0F                               mov cl,[BX]     ;get count
284D 43                                 inc BX
284E B030                               mov al, 30h
2850 803E062A00                         cmp cur_disk,0
2855 7502               2859            jne sel1        ;drive 1 if not zero
2857 B040                               mov al, 40h     ;else drive is 0
2859 A2F629             sel1:   mov sel_mask,al ; save select mask
285C 0A07                               or al,[BX]      ;merge command and drive code
285E E6A0                               out fdc_com,al  ;send command byte
                                parm_loop:
2860 FEC9                               dec cl
2862 740D               2871            jz parm_exit    ;no (more) parameters, return
2864 43                                 inc BX          ;point to (next) parameter
                                parm_poll:
2865 E4A0                               in al,fdc_stat
2867 A820                               test al,20h     ;test "parameter register full" bit
2869 75FA               2865            jnz parm_poll   ;idle until parm reg not full
286B 8A07                               mov al,[BX]
286D E6A1                               out fdc_parm,al ;send next parameter
286F EBEF               2860            jmps parm_loop  ;go see if there are more parameters
                                parm_exit:
2871 C3                                 ret

                                ;***********************************************
                                ;*                                             *
                                ;*              Data Areas                     *
                                ;*                                             *
                                ;***********************************************
  2872                          data_offset     equ offset $

                                        dseg
                                        org     data_offset     ;contiguous with code segment


2872 0D0A0D0A                   signon  db      cr,lf,cr,lf
2876 202043502F4D                       db      '  CP/M-86, 1 Feb 82 ',cr,lf,0
     2D3836202C20
     3120466562203
     3220
     32200D0A00

288D 0D0A486F6D65               bad_hom db      cr,lf,'Home Error',cr,lf,0
     204572726F72
     0D0A00
289C 0D0A496E7465               int_trp db      cr,lf,'Interrupt Trap Halt',cr,lf,0
     727275707420
     547261702048
     616C740D0A00

28B4 D428E528F628               errtbl  dw er0,er1,er2,er3
     0729
28BC 182928293529                       dw er4,er5,er6,er7
```

COPYRIGHT © 1981, 198
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 939
SER. #

```
690
691                4629
692   28C4 59296D297F29          dw er8,er9,erA,erB
693                9629
694   28CC A629BB29CC29          dw erC,erD,erE,erF
695                DD29
696
697   28D4 0D0A4572726F    er0   db  cr,lf,'Error Code 0 :',0
698                7220436F6465
699                2030203A00
700   28E5 0D0A4572726F    er1   db  cr,lf,'Error Code 1 :',0
701                7220436F6465
702                2031203A00
703   28F6 0D0A4572726F    er2   db  cr,lf,'Error Code 2 :',0
704                7220436F6465
705                2032203A00
706   2907 0D0A4572726F    er3   db  cr,lf,'Error Code 3 :',0
707                7220436F6465
708                2033203A00
709   2918 0D0A436C6F63    er4   db  cr,lf,'Clock Error :',0
710                6B204572726F
711                72203A00
712   2928 0D0A4C617465    er5   db  cr,lf,'Late DMA :',0
713                20444D41203A
714                00
715   2935 0D0A49442043    er6   db  cr,lf,'ID CRC Error :',0
716                524320457272
717                6F72203A00
718   2946 0D0A44617461    er7   db  cr,lf,'Data CRC Error :',0
719                204352432045
720                72726F72203A
721                00
722   2959 0D0A44726976    er8   db  cr,lf,'Drive Not Ready :',0
723                65204E6F7420
724                526561647920
725                3A00
726   296D 0D0A5772697465  er9   db  cr,lf,'Write Protect :',0
727                652050726F74
728                656374203A00
729   297F 0D0A547261636B  erA   db  cr,lf,'Track 00 Not Found :',0
730                6B203030204E
731                6F7420466F75
732                6E64203A00
733   2996 0D0A5772697465  erB   db  cr,lf,'Write Fault :',0
734                65204661756C
735                74203A00
736   29A6 0D0A53656374    erC   db  cr,lf,'Sector Not Found :',0
737                6F72204E6F74
738                20466F756E64
739                203A00
740   29BB 0D0A4572726F    erD   db  cr,lf,'Error Code D :',0
741                7220436F6465
742                2044203A00
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                                
29CC 0D0A4572726F       erE     db  cr,lf,'Error Code E :',0
     7220436F6465
     2045203A00
29DD 0D0A4572726F       erF     db  cr,lf,'Error Code F :',0
     7220436F6465
     2046203A00

  2959                  nrdymsg equ er8

29EE FF                 b_first_flag    db      -1

29EF 00                 rtry_cnt db 0   ;disk error retry counter
29F0 0000               last_com dw 0   ;address of last command string
29F2 0000               cur_dma_adr  dw 0       ;dma offset stored here
29F4 0000               cur_dma_seg  dw 0       ;dma segment stored here
29F6 40                 sel_mask db 40h ;select mask, 40h or 80h

                        ;       Various command strings for i8271

29F7 03                 io_com  db 3    ;length (changed to 4 to read a track)
29F8 00                 rd_wr   db 0    ;read/write function code
29F9 00                 trk     db 0    ;track #
29FA 00                 sect    db 0    ;sector #
29FB 1A                 sec_len db 26   ;transfer 26 sectors on a track read

29FC 022900             hom_com db 2,29h,0      ;home drive command
29FF 012C               rds_com db 1,2ch        ;read status command

                        ;       Track buffering variables

2A01                    cpm_disk        rb      1
2A02                    cpm_track       rw      1
2A04                    cpm_sec         rw      1

2A06                    cur_disk        rb      1
2A07                    cur_track       rw      1
2A09                    cur_sec         rw      1

2A0B                    dma_offset      rw      1
2A0D                    dma_segment     rw      1
  2A0B                  dma_longword    equ     dword ptr dma_offset

2A0F                    bdos_wr_code    rb      1
2A10                    wr_flag         rb      1

                        ;       System Memory Segment Table

2A11 02                 segtable db 2   ;2 segments
2A12 0004                       dw tpa_seg      ;1st seg starts after BIOS
2A14 0004                       dw tpa_len      ;and extends to 08000
2A16 0010                       dw 1000h        ;second is 10000 -
2A18 0010                       dw 1000h        ;1FFFF (64K)
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                        ;       include singles.lib ;read in disk definitions
                        ;               DISKS 2
  2A1A                  dpbase  equ     $               ;Base of Disk Parameter Blocks
2A1A 492A0000           dpe0    dw      xlt0,0000h      ;Translate Table
2A1E 00000000                   dw      0000h,0000h     ;Scratch Area
2A22 632A3A2A                   dw      dirbuf,dpb0     ;Dir Buff, Parm Block
2A26 022BE32A                   dw      csv0,alv0       ;Check, Alloc Vectors
2A2A 492A0000           dpe1    dw      xlt1,0000h      ;Translate Table
2A2E 00000000                   dw      0000h,0000h     ;Scratch Area
2A32 632A3A2A                   dw      dirbuf,dpb1     ;Dir Buff, Parm Block
2A36 312B122B                   dw      csv1,alv1       ;Check, Alloc Vectors
                        ;               DISKDEF 0,1,26,6,1024,243,64,64,2
                        ;
                        ;        1944:  128 Byte Record Capacity
                        ;         243:  Kilobyte Drive  Capacity
                        ;          64:  32 Byte Directory Entries
                        ;          64:  Checked Directory Entries
                        ;         128:  Records / Extent
                        ;           8:  Records / Block
                        ;          26:  Sectors / Track
                        ;           2:  Reserved  Tracks
                        ;           6:  Sector Skew Factor
                        ;
  2A3A                  dpb0    equ     offset $        ;Disk Parameter Block
2A3A 1A00                       dw      26              ;Sectors Per Track
2A3C 03                         db      3               ;Block Shift
2A3D 07                         db      7               ;Block Mask
2A3E 00                         db      0               ;Extnt Mask
2A3F F200                       dw      242             ;Disk Size - 1
2A41 3F00                       dw      63              ;Directory Max
2A43 C0                         db      192             ;Alloc0
2A44 00                         db      0               ;Alloc1
2A45 1000                       dw      16              ;Check Size
2A47 0200                       dw      2               ;Offset
  2A49                  xlt0    equ     offset $        ;Translate Table
2A49 01070D13                   db      1,7,13,19
2A4D 19050B11                   db      25,5,11,17
2A51 1703090F                   db      23,3,9,15
2A55 1502080E                   db      21,2,8,14
2A59 141A060C                   db      20,26,6,12
2A5D 1218040A                   db      18,24,4,10
2A61 1016                       db      16,22
  001F                  als0    equ     (243+7)/8       ;Allocation Vector Size
  0010                  css0    equ     16              ;Check Vector Size
                        ;               DISKDEF 1,0
                        ;
                        ;       Disk 1 is the same as Disk 0
                        ;
  2A3A                  dpb1    equ     dpb0            ;Equivalent Parameters
  001F                  als1    equ     als0            ;Same Allocation Vector Size
  0010                  css1    equ     css0            ;Same Checksum Vector Size
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                                ;
2A49                    xlt1    equ     xlt0            ;Same Translate Table
                        ;               ENDEF
                        ;
                        ;       Uninitialized Scratch Memory Follows:
                        ;
2A63                    dirbuf  rs      128             ;Directory Buffer
2AE3                    alv0    rs      als0            ;Alloc Vector
2B02                    csv0    rs      css0            ;Check Vector
2B12                    alv1    rs      als1            ;Alloc Vector
2B31                    csv1    rs      css1            ;Check Vector
                        
                        
2B41                    sec_flags       rb      host_spt
                        
2B5B                    track_buffer    rb      host_spt*host_sectsiz
                        
                        
385B                    loc_stk rw  32  ;local stack for initialization
                        
389B                    stkbase equ offset $
                        
389B                    lastoff equ offset $
                        
0400                    tpa_seg equ ((lastoff+07FFh)/0400h)*40h         ; round off to 1K boundary
0400                    tpa_len equ 0800h - tpa_seg
                        
389B 00                         db 0    ;fill last address for GENCMD
                        
                        ;*********************************************
                        ;*                                           *
                        ;* Dummy Data Section for Interrupt Vectors  *
                        ;*                                           *
                        ;*********************************************
                        
0000                            dseg    0       ;absolute low memory
                                org     0       ;(interrupt vectors)
                        
0000                    int0_offset     rw      1
0002                    int0_segment    rw      1
                        
                        ;       pad to system call vector
                                org     bdos_int * 4
                        
0380                    bdos_offset     rw      1
0382                    bdos_segment    rw      1
                        
                                END
                        
                        
END OF ASSEMBLY.  NUMBER OF ERRORS:   0.  USE FACTOR: 13%
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
ALS0            001F N   840#  847   856
ALS1            001F N   847#  858
ALV0            2AE3 V   804   856#
ALV1            2B12 V   808   858#
BADMON          288D V   589   679#
BASE204         00A0 N   47#   49    50    51    52    53    55    56    57    58
                         59    60    61    62    63
BDOSINT         00E0 N   23#   891
BDOSOFFSET      0380 V   126   893#
BDOSOFST        0B06 N   31#   126
BDOSSEGMENT     0382 V   894#
BDOSWRCODE      2A0F V   736#
BFIRSTFLAG      29EE V   460   463   753#
BIOSCODE        2500 N   29#   73
BLCRESET        0060 N   38#   131
CCP             0000 L   72#   144   146
CCPOFFSET       0000 N   30#   71
CDATA           00D8 N   34#   178   187   528
CLEARFLAGS      270E L   141   429#
CONIN           25A5 L   84    175#  177   228
CONOUT          25AF L   85    182#  185   231   248
CONST           259C L   83    167#  176
CONSTRET        25A4 L   170   172#
CORRECTTRACK    26C4 L   369   388#
CPMDISK         2A01 V   261   363   381   774#
CPMSEC          2A04 V   288   389   776#
CPMTRACK        2A02 V   284   367   379   775#
CR              000D N   67#   673   673   674   679   679   682   682   697   700
                         703   706   709   712   715   718   722   726   729   733
                         736   740   744   747
CSS0            0010 N   841#  848   857
CSS1            0010 N   848#  859
CSTS            00DA N   33#   168   183
CSV0            2B02 V   804   857#
CSV1            2B31 V   808   859#
CURDISK         2A06 V   278   364   382   435   458   642   773#
CURDMAADR       29F2 V   383   417   629   757#
CURDMASEG       29F4 V   442   633   758#
CURSEC          2A09 V   385   410   466   780#
CURTRACK        2A07 V   368   380   468   779#
DATAOFFSET      2B72 N   667#  670
DIRBUF          2A63 V   803   807   855#
DMACADR         00A4 N   55#   630   632
DMACCONT        00A5 N   56#   626   628
DMACMODE        00A8 N   59#   624
DMACSADR        00A7 N   58#
DMACSCAN        00A6 N   57#
DMACSTAT        00A8 N   60#
DMAEXIT         284B L   618   638#
DMALONGWORD     2A0B V   323   338   784#
DMAOFFSET       2A0B V   294   782#  784
DMASEGMENT      2A0D V   297   783#
DPB0            2A3A N   803   821#  846
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
USER. #_____

```
DPB1            2A3A N   807   846#
DPBASE          2A1A L   271   800#
DPE0            2A1A V   801#
DPE1            2A2A V   805#
DRNRDY          27CC L   512   541#
DRPOLL          27FA L   575#  578
DRRDY           27A2 L   514#
ER0             28D4 V   687   677#
ER1             28E5 V   687   700#
ER2             28F6 V   687   703#
ER3             2907 V   687   706#
ER4             2918 V   689   709#
ER5             2928 V   689   712#
ER6             2935 V   689   715#
ER7             2946 V   689   718#
ER8             2959 V   692   722#  751
ER9             296D V   692   726#
ERA             297F V   692   729#
ERB             2996 V   692   733#
ERC             29A6 V   694   736#
ERD             29BB V   694   740#
ERE             29CC V   694   744#
ERF             29DD V   694   747#
ERRTBL          28B4 V   526   637#
EXECEXIT        27CB L   497   509   538#
EXECPOLL        2790 L   492   499#  504
EXECUTE         2767 L   473#  587
FALSE           0000 N   20#
FDCCOM          00A0 N   49#   647
FDCPARM         00A1 N   51#   657
FDCRSLT         00A1 N   52#   507   579
FDCRST          00A2 N   53#
FDCSEGMENT      00AA N   62#   634   636
FDCSEL          00A9 N   61#
FDCSTAT         00A0 N   50#   501   576   605   653
FLUSHBUFFER     26D3 L   353   377   399#
GETIOBF         25D2 L   100   212#
GETSEGT         2645 L   99    311#
HOMCOM          29FC V   586   769#
HOME            2615 L   89    275#
HOME1           2621 L   276   280#
HOSTFSN         0001 N   26#   385   390   391   402   405   407   412   413
HOSTSECTSIZ     0080 N   24#   343   411   864
HOSTSPT         001A N   25#   403   440   862   864
INIT            253F L   81    106#
INITDMA         282D L   615   620#
INTOUFFSET      0000 V   119   887#
INTOSEGMENT     0002 V   120   888#
INTTRAP         2590 L   119   149#
INTTRP          289C V   153   682#
IOCOM           29F7 V   449   450   454   455   465   763#
LASTCOM         29F0 V   479   484   488   756#
LASTOFF         389B N   871#  873
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER #_____

```
LDATA           0040 N    37#  133  135  137  195
LF              000A N    68#  673  673  674  679  679  682  682  697  700
                          703  706  709  712  715  718  722  726  729  733
                          736  740  744  747
LISTOUT         25BA L    86  190#  193
LISTST          25C4 L    96  192  198#
LOCSTK          385B V   867#
LSTS            0041 N    36#  200
MAXRETRIES      000A N    65#  482
NEXTSECT        26E0 L   404#  423
NOFLUSH         2708 L   401   424#
NOTFIRSTB       2758 L   458   460  464#
NOTUPDATED      2705 L   406   421#
NRDY01          27DC L   548#  550
NRDY2           27F4 L   570   572#
NRDYMSG         2959 V   546   751#
OUTERRETRY      276B L   480#  534
PARMEXIT        2871 L   650   659#
PARMLOOP        2860 L   648#  658
PARMPOLL        2865 L   652#  655
PMSG            25EE L   139   154  243#  250  527  547  590
PUNCH           25CF L    87   207#
RDSCOM          29FF V   573   770#
ROWR            29F8 V   764#
READ            2649 L    94   317#
READER          25CF L    88   208#
RESET204        00AF N    63#
RESTORE         2805 L   462   516  585#  591
RESTOREEXIT     2815 L   588   592#
RETRY           2770 L   483#  519  543   545  551
RETURN          2614 L   246   264  272#
RTRYCNT         29EF V   482   518  755#
RWCOMMON        2742 L   451   457#
SECFLAGS        2841 V   347   405  407   438  862#
SECLEN          29FB V   767#
SECT            29FA V   467   766#
SECTORWRITE     2738 L   418   453#
SECTRAN         2638 L    97   300#
SECTRANEXIT     2644 L   303   307#
SEGTABLE        2A11 V   312   791#
SEL1            2859 L   643   645#
SELDSK          25FC L    90   260#
SELMASK         29F6 V   569   645  759#
SENDCOM         2816 L   485   574  604#  607
SETDMA          262E L    93   293#
SETDMAB         2633 L    98   298#
SETIOBF         25D5 L   101   217#
SETSEC          2629 L    92   287#
SETTRK          2624 L    91   283#
SIGNON          2872 V   138   673#
STKBASE         389B N   112   869#
TESTREADY       27E9 L   542   544  549   567#
TPALEN          0400 N   793   874#
```

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

TPASEG	0400	N	792	873#	874			
TRACKBUFFER	285B	V	321	336	383	416	864#	
TRACKREAD	272C	L	386	448#				
TRACKSETUP	2694	L	318	331	361#			
TRK	29F9	V	469	765#				
TRUE	FFFF	N	19#	20				
UCONECHO	25D9	L	227#	529				
URET	25ED	L	234	236	238#			
WBOOT	258D	L	82	146#	557			
WBOOTL	27E6	L	532	556#				
WRFLAG	2A10	V	276	348	400	426	437	787#
WRITE	265C	L	95	329#				
WRITENAYBE	2827	L	613	616#				
WRITERETURN	2691	L	351	356#				
WRONGTRACK	26A6	L	365	376#				
XLT0	2A49	N	801	832#	850			
XLT1	2A49	N	805	850#				
ZERORET	25D6	L	203	220#				
ZRET	27E3	L	536	552#				

COPYRIGHT © 1981, 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____